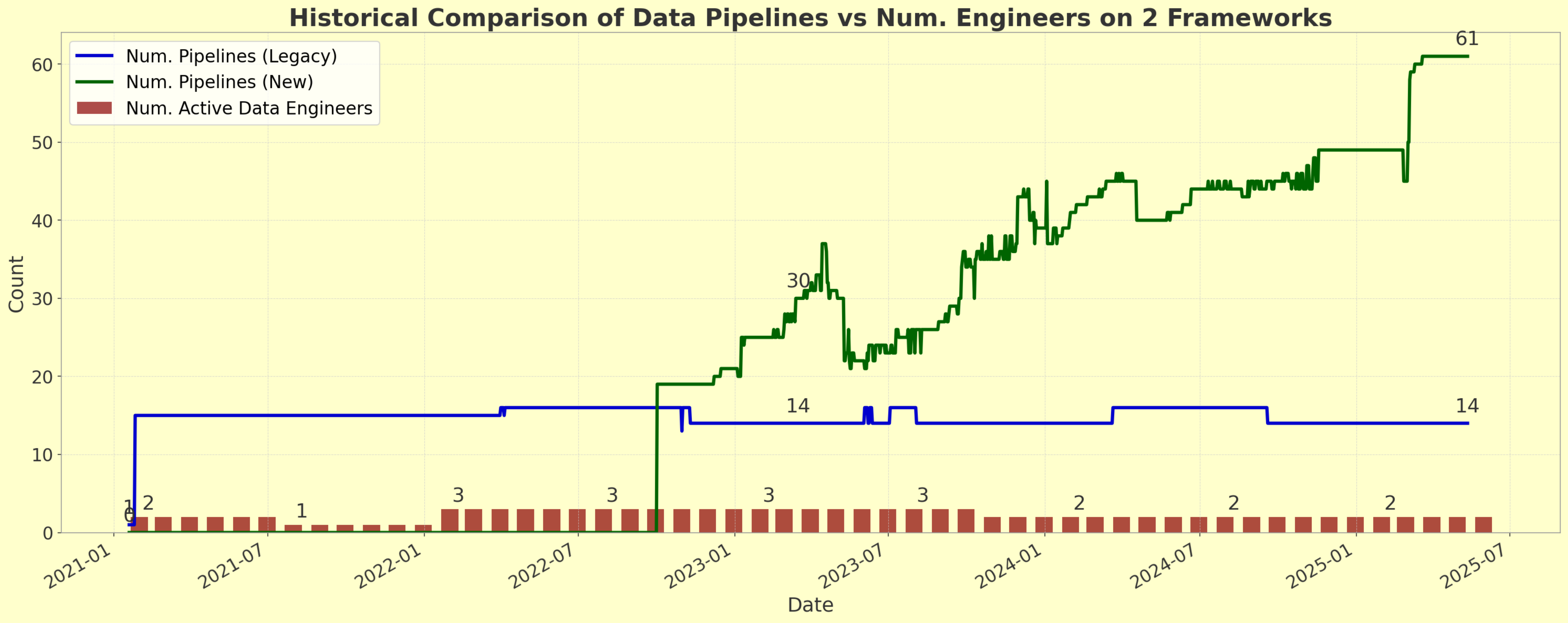
Bassoon: A Low-Code Data Engineering Framework

By Yueyang Chen

A straightforward view to the benefits

1. Rapid growth of new data pipelines despite barely new data engineers
2. Nike-wide system outages, software upgrades, and data migrations have little impact to the growth curve



Why does bassoon work?

The theory

1. Complex data sources — data engineering —> complex business requirements
2. Maintenance cost = data complexity x business complexity x (business code + engineering code)
3. Therefore to maintain business running, the only approach is to reduce engineering code

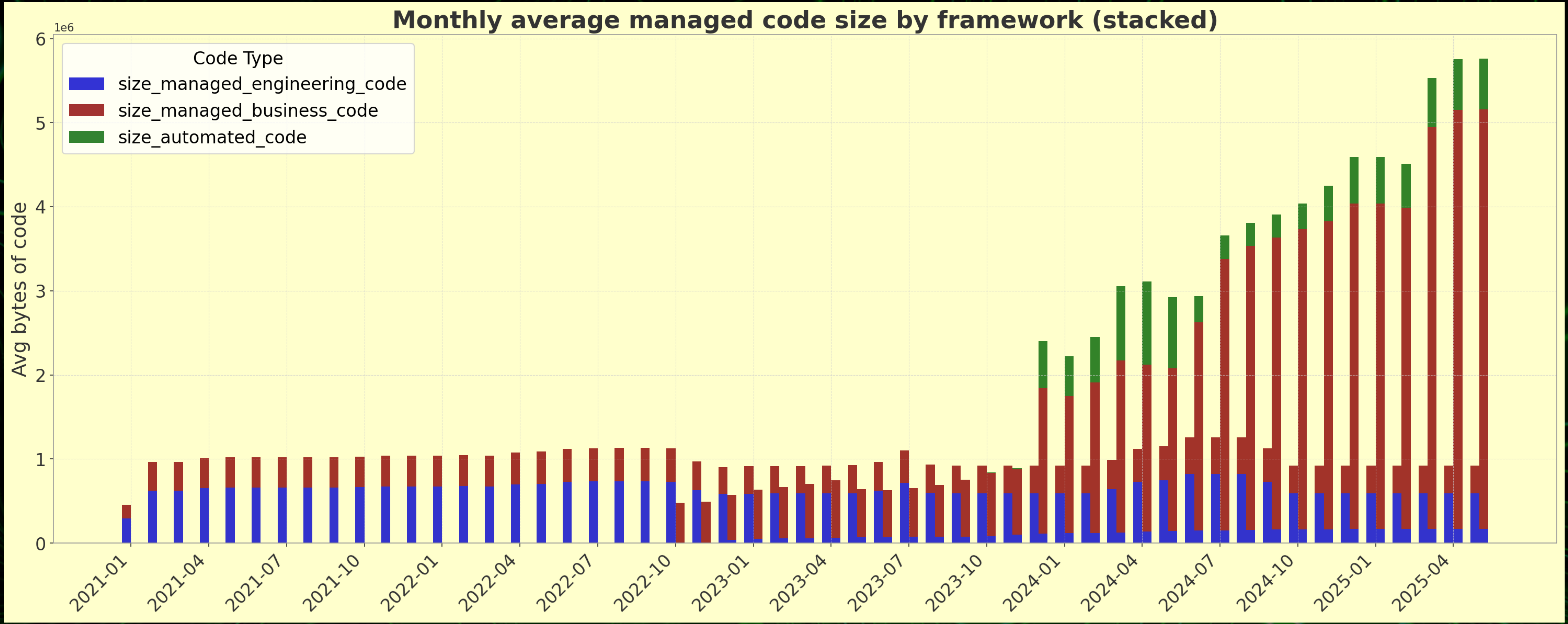
Common pitfalls of existing frameworks

1. Heavy reliance on third-party platforms (AWS/Azure), leading to lock-in
2. Tightly coupled to certain data, business specifics, or platforms, limiting versatility.
3. Blurred lines between engineering and business code, complicating development and excluding non-technical particip

The solution

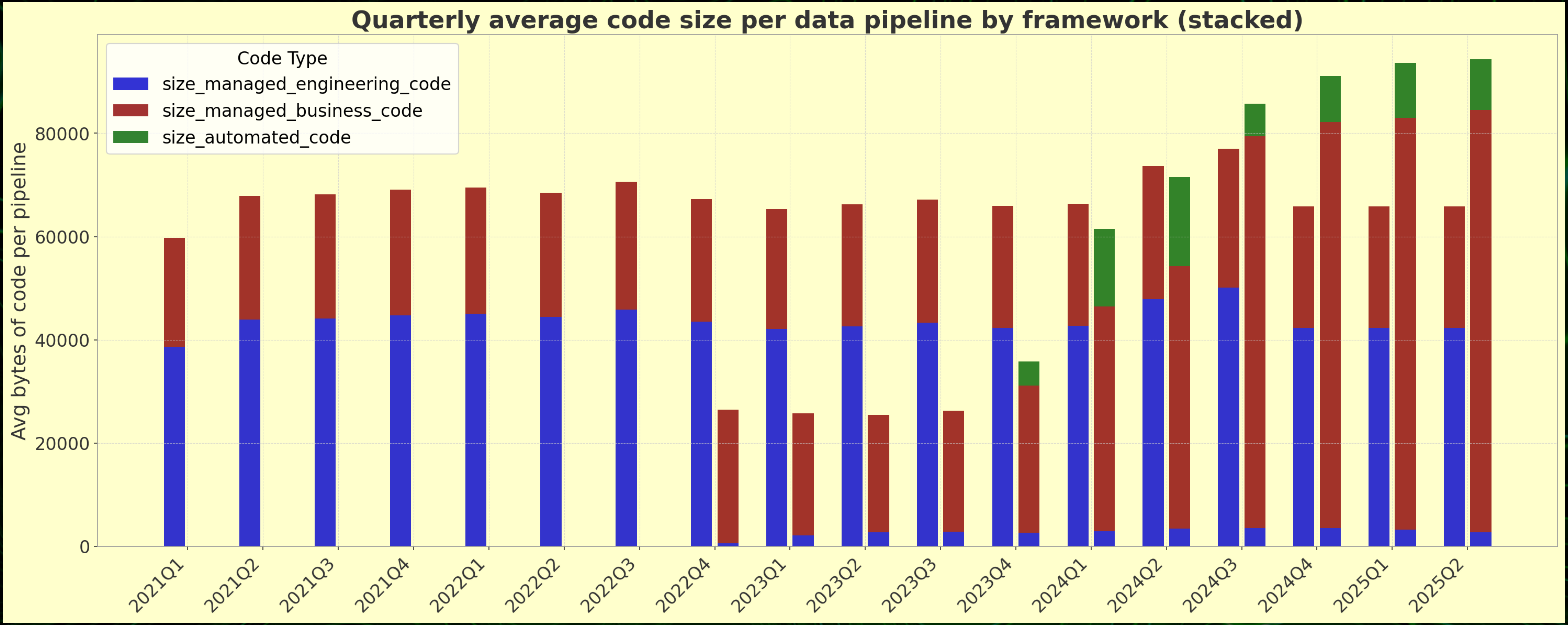
1. Let data engineers focus solely on business logic
2. Clearly defined rules cover data processing and platform constraints.
3. Auto-generated engineering code

Bassoon enabled engineers to handle more data pipelines



1. Pre-bassoon: blue bars squeezing red bars, meaning that engineers bogged down by maintaining engineering code
2. With-bassoon: red bars squeezing blue bars with rapid growth, meaning (a) reduced engineering workload (b) more people participated in engineering

Bassoon enabled engineers to handle more complex logic



Bassoon keeps business running during disruptions

- On 2025-02-21, an AWS-related failure rendered some legacy distributed machines unusable
- Within **two hours**, we not only identified the root cause but also migrated a large amount of code to newer machines to keep things running—without even triggering alerts downstream.
- In contrast, AWS didn't resolve the issue with those old machines until **10 days later**.

- In Q1 2023, Nike Global Tech required all teams to upgrade Airflow, a company-mandated data tool. The upgrade was was initially expected to consume significant engineering resources. However, we completed it within **a month**.
- In contrast, one of our peer teams didn’t announce their completion until **8 months** later.

- In Q2 2024, a downstream team strongly requested access to review parts of our data pipeline code. Due to privacy regulations, sharing will cost significant engineering effort.
- However, we completed the migration in just **two days** and kicked off several months of deep, technical collaboration with them.

Why did I make bassoon

- Experienced intense technological upheavals: During my tenure at Boston Consulting Group (New York), I participated in digital transformation projects for industry giants in various sectors. The team had originally developed an in-house ETL framework, but for every client served, this framework either relied too heavily on AWS or clashed with the client's outdated tech stack, rendering it unusable. Each time, we had to hastily build a new framework from scratch, barely meeting deadlines.

- Witnessed the horrors of poorly engineered ETL: Two clients initially handed their projects to Data Scientists for POC development before transferring them to us for business implementation. After refactoring 6,000 lines of PySpark code, I came to deeply understand the significant communication gap between engineering and non-engineering teams in data projects.

- The same issue resurfaced when I joined Nike. Driven by past frustrations, I began developing and promoting Bassoon within the team starting July 2022